Software Requirements Specification

for

<Project>

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Person's Name, not "Company X". Multiple people are okay if they all worked on this version			

1. Introduction

1.1 Purpose

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

This section describes the software very briefly, and notes if it is the whole system or part of a larger system. Very short.

1.2 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Any external document, specifications? If not, say "None."

2. Overall Description

2.1 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

You must have at least two for CS421. This could be regular and administrator, power user, etc…

2.2 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

This should be short, and mainly include what types of environments you support. If it's a web-based product, this is browsers and versions. (Don't say "all browsers". There are hundreds of web browsers… SPECIFICALLY list which ones you will support.)

If it's a (non-web) application, provide the environment: Windows Vista on a P4 100MHz and higher. Look at software specifications on the web, and see what they say.

2.3 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

You may have some or None. Think about it though, if you write "none", but it is obvious you should have some, I will deduct points!

2.4 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

You may have some or None. Think about it though, if you write "none", but it is obvious you should have some, I will deduct points!

3. External Interface Requirements

SKIP ALL OF SECTION 3 for the SRS… We'll do this as part of the GUI Prototype

3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

Think about this in terms of standards, not specific features. For example,

- *all buttons will have a black border*
- *all fonts will be Arial*
- *Draw a screen template showing "main area", "menu here", "status bar" and describe each component. If you have multiple screen layouts depending on the user's current task/settings, describe them*
- *What screen resolutions will you support?*
- *Will you be Section 508 compliant? Are there any other standards you support?*
- *etc…*

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

If you system doesn't include hardware, then you'll have none. If it has hardware components, then you should describe (at a high level) how you interface with that hardware.

3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

These are internal connections to things like databases, web servers. You mainly need to explain that you have them, but (for CS421) I don't expect detailed information about how you actually connect to them and use them. Just explain that you ***do*** *connect to them and use them for storage of customer information, or to process incoming web requests, etc… These are internal components of your system.*

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

These are external communication mechanisms. Do you connect to a bank computer to verify credit card information? That is NOT part of your system, so it is an external communication you have. Describe it here. Do you have other systems connecting in to yours to perform some function? That would also go here.

4. System Use Cases

The overall use case diagram should be here.

The text description of each use case should follow.

4.1 Use case name and identifier

1. **Unique Identifier**
2. **Objective** - What is the ultimate objective of the use-case. What is it trying to achieve? What was the source of the use-case requirement?
3. **Priority** – The overall priority of this use-case (Low, Medium, High)
4. **Source** – Who is the main source of this use case. Who cares most about this functionality? This should be the one person you would ask if there is a question about this use-case. (Make up a name and cite their: John Smith (End-user) here.)
5. **Actors** - Who is involved in the use-case? Which actors/stakeholders?
6. **Flow of Events**
   - 6.1. **Basic Flow** - flow of events normally executed in the use-case
   - 6.2. **Alternative Flow(s)** - a secondary flow of events due to infrequent conditions
   - 6.3. **Exception Flow(s)** - Exceptions that may happen during the execution of the use case
7. **Includes** - other use case IDs that are referenced in steps in the flow of events.
8. **Preconditions** - Any condition that must be satisfied before the use case begins. If the condition is "User is logged in", then the first step of the use case is NOT "User logs in". They are already logged in if that is a pre-condition!
9. **Post conditions** - The conditions that will be satisfied after the use case successfully completes
10. **Notes/Issues** - Any relevant notes or issues that need to be resolved

4.2 Withdraw money from ATM (U2)

1. **U2**
2. **Objective** – The customer is withdrawing money from the ATM and the system will debit the customer's account.
3. **Priority** – High
4. **Source** – Carl Gnome (marketing)
5. **Actors** – Customer, central bank computer
6. **Flow of Events**
   - 6.1. **Basic Flow**
     - 6.1.1. Customer chooses the checking option on the ATM
     - 6.1.2. Customer chooses the amount of money needed
     - 6.1.3. Customer confirms the choice
     - 6.1.4. System validates the amount
     - 6.1.5. System asks central bank computer to debit the customer's account
     - 6.1.6. System issues money to the user
   - 6.2. **Alternative Flow 1** – At step 5.1.4 the amount is not a multiple of $20
     - 6.2.1. An error message is displayed telling the customer they must use multiple of $20.
     - 6.2.2. Return to step 5.1.2
   - 6.3. **Alternative Flow 2** – At any step the user presses "cancel"
     - 6.3.1. System returns to the main menu
   - 6.4. **Alternative Flow 3** - At step 5.1.5 bank computer returns a failed status, "insufficient funds"
     - 6.4.1. An error message is shown to the user
     - 6.4.2. Return to step 5.1.2
   - 6.5. **Exception Flow 1** –
     - 6.5.1. Database is locked due to backup in progress. System executes use case U5
7. **Includes**

7.1. U5 – Exception occurs
8. **Preconditions** – User is logged in
9. **Post conditions** – Money has been returned to the user and their account balance has been updated.
10. **Notes/Issues** - None

4.3 Deposit money into ATM (U3)

….

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

In this section, just say "See section 7 requirements 23-27". And I'll assume those requirements are Performance related.

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

In this section, just say "See section 7 requirements 25-32". And I'll assume those requirements are Safety related.

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

In this section, just say "See section 7 requirements 35-42". And I'll assume those requirements are Security related.

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

In this section, just say "See section 7 requirements 55-62". And I'll assume those requirements are Software Quality related.

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

You may not have any.

7. System Requirements Chart

*< Include a **table** in this section with the following columns:*

***ID** – Unique requirement ID*
***Priority** – Priority of this requirement*
***Type** – Functional(F) or Non-functional(NF)*
***Source** – Who is most interested in this requirement (John Smith – Customer). For this project you can make it up, in reality you'll want to capture this as you capture the requirements.*
***Contained in Use Case(s)** – Which use cases reference this requirement or which use cases when executed will perform this requirement. There may be a few functional requirements without a use-case and the non-functional requirements generally will NOT be part of a use-case (so put N/A).*
***Description** – The description of the requirement. "The system shall …. "*
>

These requirements should match up with your use case diagrams.

Appendix A: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Don't do any of these for CS421 SRS. You will create these models during the high level design deliverable.

Appendix B: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

List here any open questions or things you know still need to be done to the SRS, but haven't been addressed yet. (It's okay to have things like that, especially in this CS421 project because we don't have time to do everything.)